BLINK: Pixel-Domain Encryption for Secure Document Management

Idris Atakli
Department of Electrical and Computer Engineering
Binghamton University
Binghamton, NY
iatakli@binghamton.edu

Yu Chen
Department of Electrical and Computer Engineering
Binghamton University
Binghamton, NY
ychen@binghamton.edu

Qing Wu
Department of Electrical and Computer Engineering
Binghamton University
Binghamton, NY
qwu@binghamton.edu

Scott Craver
Department of Electrical and Computer Engineering
Binghamton University
Binghamton, NY
scraver@binghamton.edu

ABSTRACT

BLINK, or brief-lifetime ink, is a technology for secure document delivery and management that employs pixel-domain scrambling of raster images by a hardware device connecting a computer to its display. A BLINK decoder monitors digital video signals, identifies the presence of scrambled images, and selectively decrypts regions of the video frame containing them. The primary application is delivery of confidential documents that can only be viewed by a specific machine or user, possibly within a fixed time limit or for a fixed number of views.

Removing the decryption step from the computer to the display cable confers numerous advantages, including complete protection against document forwarding, copying, pasting, screen capture, and memory snooping; furthermore it requires no particular operating system, reader software or proprietary document format.

In this paper we implement several core BLINK primitives on an Alteral FPGA development board. These primitives include encrypted image identification and location, key extraction, decryption by a key stream, and bit-plane extraction for encrypted images embedded in LSB planes of other images.

Categories and Subject Descriptors

H.4 [**Information Systems Applications**]: Miscellaneous; K.6.5 [**Management of Computing and Information Systems**]: Security and Protection—*Authentication, Physical Security*

1. INTRODUCTION

A daunting task in any organization is the management of confidential material in the digital age. If one distributes a digital document via email or within an internal network, it is easily leaked by numerous mechanisms: one can store a document on a thumb drive, forward an email, and copy and paste text. If these are prevented one can always capture the user's screen and steal that bitmap file. The threat is not limited to an insider seeking to leak confidential information; malicious code can also capture a user's screen or memory.

Ideally, one would like to distribute a confidential document to a group and be guaranteed control over its dissemination and use. This would include limiting the specific machines or users able to view the document, as well as limiting the time frame during which a document is readable. Such "self-destructing" email is difficult to implement because it typically requires cooperation of a user's computer, including the use of customized software or a patched operating system.

BLINK, or *brief-lifetime ink,* is a form of secure document distribution that employs raster images encrypted in the pixel domain, and decrypted in real time by a dedicated device on the cable between the user's computer and display. A BLINK object is an image whose top raster contains an identifier and header that identifies a region of the screen as encrypted with a given initialization vector, to be viewable within a given timeframe. A rudimentary description of the specification is outlined in [4].

In deployment, a BLINK decoder would simply be a video cable with an integrated decoder box; its interface is a single switch by which the user can manually switch its behavior between pass-through mode—in which the box behaves like a regular video cable—and secure decoding mode. In secure mode, the decoder is able to identify BLINK objects on screen by identifying their raster lines in the serial video data; it can then selectively descramble the region of the display where the image resides. A key infrastructure would then allow the image to be selectively viewable by an in-

dividual machine or user, as well as retire document keys after a document's lifetime is passed. This provides a secure mode of document delivery that can render a document unreadable after a fixed time or number of views.

1.1 Advantages of pixel-domain document encryption

The chief innovation behind BLINK is the rendering of documents as multimedia objects, specifically raster images, so that they can be decrypted outside of the domain of the computer. This confers numerous advantages over other means of secure delivery:

- The encrypted document is never decrypted on the user's machine. Thus there is no plaintext version to be copied, pasted, forwarded, or even captured by a print-screen operation.
- Blink documents can be embedded in common formats and viewed by existing applications. There is no need for a customized secure email viewer or other restrictive software to deliver messages with forwarding restrictions. Requiring a customized email or document reader program is both a violation of Kerckhoffs's Criterion, as well as the general usability principle that users tend to circumvent security measures that are sufficiently inconvenient.
- Restrictions on copying, pasting, and screen capture do not require a speclialized or patched operating system. Requiring a specific operating system or patched version thereof is not only restrictive, it can result in poorer security overall if a necessary operating system update would break a patch.
- Key management and access control rules are maintained on dedicated hardware, which can be managed by a central authority and secured using physical tamper resistance or tamper evidence techniques. For example, in a secure installation, users' computers can be fit with individually keyed BLINK cables, secured to desktops, requiring a physical connection to a computer or display to maintain battery power to stored keys.

This paper is organized as follows. In section 2, we discuss the overall BLINK architecture. In section 3, we describe our current implementation of a corse subset of BLINK features. In section 4, we show our existing results and current design challenges.

2. THE BLINK ARCHITECTURE

At its most abstract, BLINK is an extensible standard for extracting and altering raster information from specialized images on a display. Alterations are not limited to descrambling, but can also include steganographic decoding, display watermarking, and authentication of users.

2.1 BLINK images

A BLINK image is a possibly encrypted image whose first and last rows contain metadata regarding the encryption method. Because of the serial nature of the digital display signal, it is crucial that information can be decoded easily by horizontal scanlines. It is similarly important for the encryption itself to be compatible with horizontal image scanlines.

The first row of each image is a registration pattern which is detected onscreen and used to determine the horizontal and vertical position of an encrypted image. Short registration patterns can also be placed in the left and right margins of the image for convenience in decoding. After the image is a header that contains metadata encoded as name-value pairs. This can include initialization vectors for decoding, as well as instructions for what type of decoding is necessary. An *encrypted footer* is placed in the bottom border, allowing the sending of name-value pairs that cannot be easily altered. This can include usage restrictions on the image.

BLINK images do not need to contain encrypted data. One other application is communication with the device for configuration or authentication.

2.2 Decoding

A BLINK decoder is essentially a DVI display cable with a decoder in the middle. By default the decoder acts as a buffer, passing the input stream to the output. When a button is pressed on the decoder unit, it monitors the serial pixel data for evidence that a BLINK image is onscreen. When an image is found, information is taken from its header and used to selectively decrypt the portion of the screen where the image resides.

The decoder contains a store of key data that can be charged by a central authority. BLINK images can be targeted to a specific decoder by encrypting with one of the decoder's stored keys. The specific key management regime will depend on the type of document infrastructure needed. In all cases the image should be encrypted with a session key, whose encryption with the decoder key is placed in the header. This allows a document to be viewable by multiple machines and in multiple circumstances.

2.3 Modes of operation

A decoder should be capable of at least several useful operations on serial pixel data.

- Decryption: an initialization vector in the image header is used to seed a cipher in counter mode.
- Bit-plane exclusion: as a form of cheap steganography, a header command can direct the decoder to discard the m most significant bits of each pixel, shifting the remaining bits to the left. This allows the embedding of binary images as ciphertext within cover images.

  While this is trivially detectable, it can allow a BLINK document to appear innocuous under visual inspection. More usefully, this allows encrypted documents to have a readable "cover page" that marks a document as confidential.
- Screen watermarking: one possible attack on the BLINK architecture is that of photographing the display. While little can be done to prevent this form of screen capture, screen watermarking may allow a captured image

to be traced to an individual decoder. When watermarking is enabled, the complement of the image is filled with a pseudo-random pattern specific to the decoder. This can act as a visible watermark to identify the machine from which a photograph was taken; an invisible robust watermark is also possible, but realistically an adversary could defeat either by passing a confidential document through optical character recognition.

- Key indirection: if multiple users share a machine, there is a simple trick for targeting a document to a specific user. A BLINK document can instruct the decoder to extract some necessary key information from a second image onscreen. This allows a central authority to issue each user a "security badge" in the form of an image, which must be onscreen for a document to be readable.

2.4 Viewing and date restrictions

A major application of BLINK is sending time-limited or limited-use documents. For example, a user can be sent a document that becomes unreadable at the end of the day, either because the decoder is directed not to decrypt or, more sensibly, because the decoder loses its key information at which point it is genuinely unable to decrypt.

The appropriate method of date or usage restriction depends on the key management regime. The most flexible form of date/usage restriction is to provide a set of viewing rules in the encrypted footer. These can instruct the device to refuse encryption when the date is not appropriate.

A second method of date restriction is key retirement: a stored key can simply be lost after a BLINK image is decoded. For example, a target decoder can decode a BLINK image with its stored key, and after the screen is switched to a normal display the key is replaced with a hash of itself. From that point onward the user is no longer able to display the image. Alternately, a key can be replaced with a hash of itself every day, allowing a document to be viewable within a one-day window, or multi-day window if multiple encryptions of the session key are placed in the header.

3. IMPLEMENTATION

We implemented a BLINK decoder on an Altera Cyclone III FPGA development board with a Bitec DVI HSMC daughtercard, shown in figure 1. The HSMC daughtercard inputs and outputs serial DVI signals, making each 24-bit pixel available for processing at the display's character clock rate [3].

For our implementation, we used Altera Quartus II software (ver 8.1), SoPC (System on a Programmable Chip) and VIP megacore functions included in SoPC libraries. We built our design on top of a basic video processing pipeline which consists of DVI Input component, Frame Buffer component, and DVI Output component. DVI Input component takes a 1024x768 DVI video signal from the HSMC DVI Input port and converts it into Avalon Data Streaming Protocol developed by Altera [1]. Frame Buffer component stores video packets on the off-chip DDR SDRAM. DVI Output component converts the processed video from Avalon Protocol back into DVI Protocol.

We built our custom Descrambler component in VIP megacore functions library and added this component in between DVI Input component and Frame Buffer component. The Descrambler passes through all the video packets except the ones which are in the detection region. The ones in detection region are descrambled using the method depicted in Figure 3 and passed to the frame buffer.

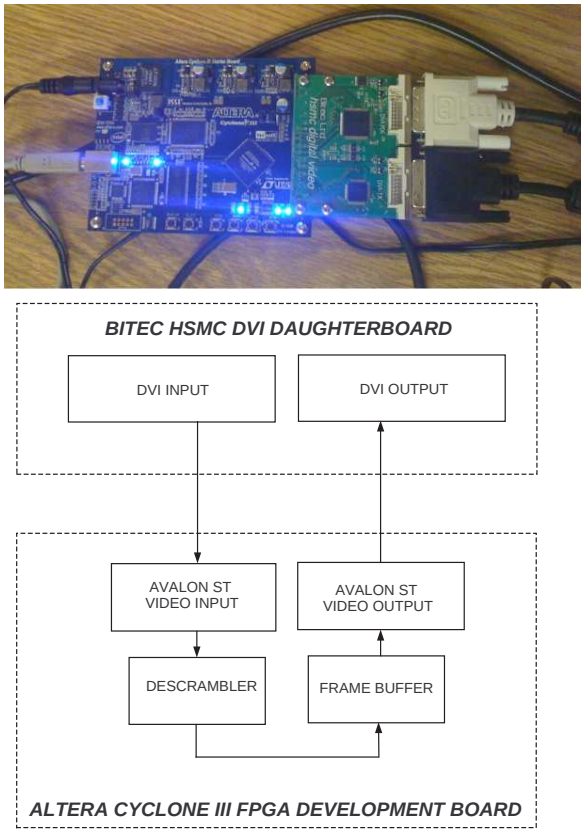

Figure 1: Above, the Altera Cyclone III development board with DVI daughtercard. Below, the block diagram of the board, DVI daughtercard and descrambler.

The logical dataflow of our decoder is illustrated in figure 3. We have coded a bit shift unit and a rudimentary decrypt unit. Decryption uses a seed that can be extracted as an initialization vector from the BLINK image header. Bit shifting allows a cover image to be placed atop the message.

The DVI standard uses special character codes to mark the end of a raster line or frame [5]. These are decoded by the HSMC daughtercard to a row and frame clock; the pixel clock and row clock can be used to determine the location of an encrypted image when its registration pattern is detected. This is used to guide the stream-cipher decryption of the image region and to synchronize the stream PRNG.

4. RESULTS

Figure 4 shows a computer display with a BLINK image, and the same display with image decoding.

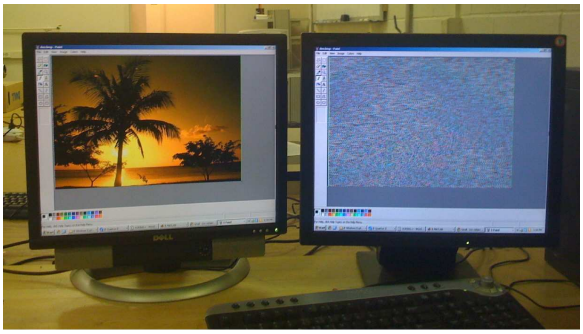

Figure 2: Our experimental setup, showing the video signal before (left) and after (right) passage through our descrambler.

We found that registration pixels along the left margin was necessary to reliably synchronize the image. We use a rudimentary PRNG for our initial experiments—a linear feedback shift register—which does not provide security. While our Cyclone III board has clock limitations preventing our decoding at higher resolutions, this does not prohibit the implementation of a strong cipher stream. The rate-determining step of the board is simply processing pixels at the character clock rate; the computation of a PRNG stream is easily parallelized on the FPGA hardware.

4.1 LSB embedding in cover images

Figure 5 illustrates a BLINK image placed in the LSB plane of a second image. The image before no longer resembles a random bit plane. While this may seem like an obscurity tactic, its main application is placing instructions or warnings in the image to let the viewer know that it is confidential and not to be disseminated.

4.2 Technical issues

The resolution of the display is limited by the reliable clock rate of the FPGA device. The development board is able to process 1920 x 1200 resolution however our design runs on 1024x768 resolution. In early experiments, higher resolutions result in skipped pixels that accumulate to produce a noisy display in the lower half of the screen. Future work may require a faster board.

Another issue to is the need for a frame buffer in our design because image buffering is introduced in many video processing pipelines in which the input and output images are unsynchronized. Before adding the frame buffer, we were not successful in doing the descrambling operation on the fly so we switched back to our original design. The remainder of the implementation is designed so as to process the entire video signal without buffering; unfortunately, it seems that buffering, while wasteful, is ultimately necessary.

5. DISCUSSION AND CONCLUSIONS

We have implemented some of the encryption primitives of the BLINK architecture, exhibiting a proof-of-concept decoder that can decrypt a document between computer and display. This decoder is capable of descrambling and bitplane shifting, performing both rudimentary steganography and decryption.

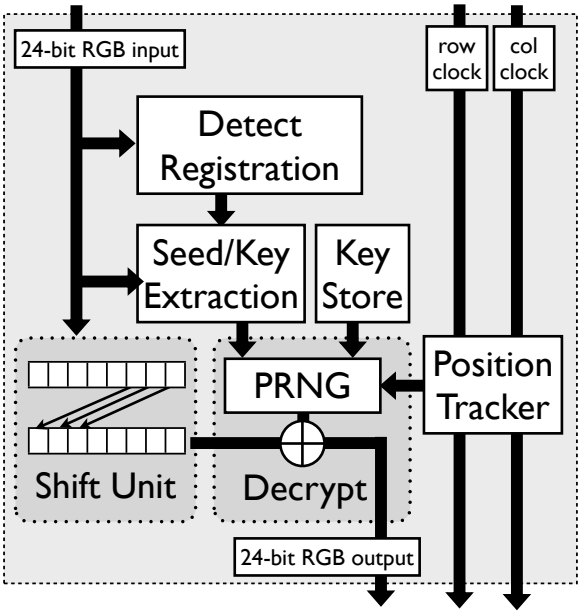

Figure 3: The decoder block diagram. Deserialized pixel data is passed through a BLINK header detector, which triggers selective shifting and scrambling. When detection is disabled, the decoder acts as a pass-through device.

The original problem we set out to solve was the transmission of time-limited self-destructing email documents. While this was not originally a "multimedia" problem, we quickly determined that the best solution was to render documents as computer images that can be amended in display.

We feel this is the most comprehensive and practical means to implement site-wide document security. Aside from being OS-independent, application neutral *and* format neutral, access controls can be managed explicitly by a central authority managing physical cables. This allows an explicit chain of custody that is harder to achieve in software. We envision a secure installation in which an administrator programs and issues BLINK decoder cables, physically securing them to a desktop and storing their public and secret keys. The adminisrator can then act as a trusted third party, accepting documents in various formats and routing them to recipient machines or users.

Since decryption takes place on a physical device, various tamper-evidence techniques can be employed. One obvious measure is using a physical switch or pin to cut power from the device's battery to the key memory if the cable is unplugged from the computer or monitor. Such methods of tamper prevention are not perfect, as various techniques have evolved to revive key data due to memory remanence. [2] A more useful feature may simply be an alarm that sounds when a device is unplugged.

5.1 Future Work

Our obvious next step is choosing and implementing a proper secure stream cipher. We also plan to implement key redirection, so that the presence of one image on the screen (a

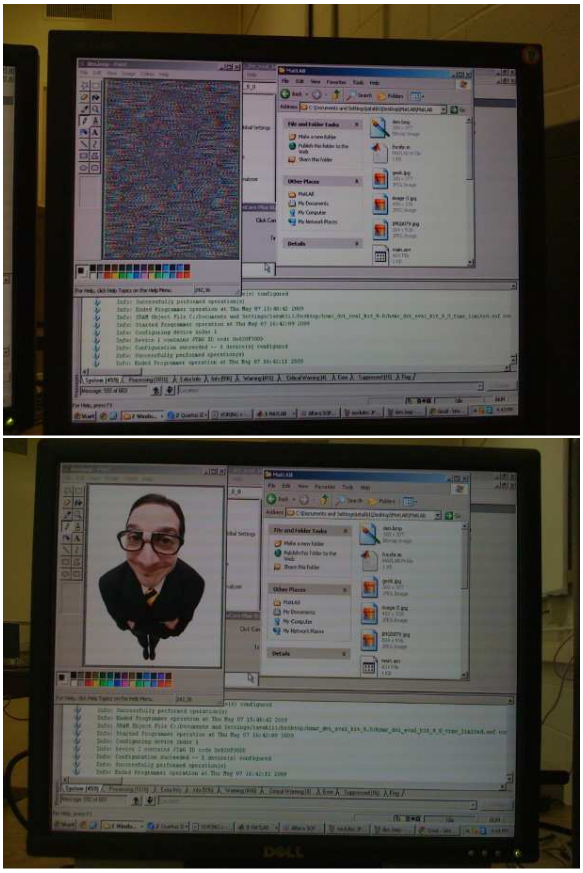

Figure 4: A BLINK image on screen, before and after passage through the decoder cable.

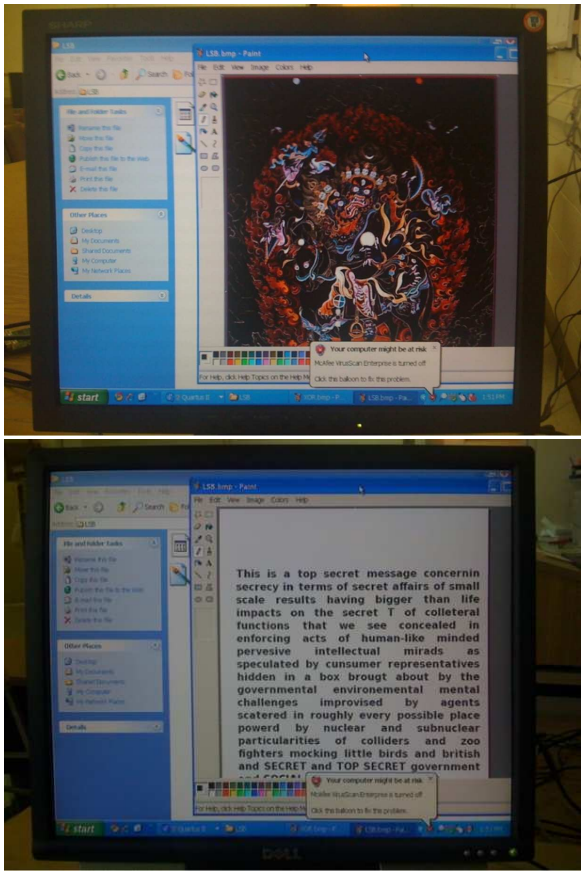

Figure 5: Bit shifting in action. This allows rudimentary LSB embedding in BLINK objects.

user's "security badge") is required for decryption of a second image. Faster hardware will allow higher resolutions. These, however, are straightforward implementations of our existing design. The major development step at this point is the design of a key management regime that will allow a central authority to flexibly dictate usage rules and time limitations on encrypted documents.

It may seem obvious to include a public-key infrastructure on BLINK decoders; however, the architecture already assumes and indeed requires a trusted third party who securely assigns keys, and physically assigns and confiscates encryption devices. Public key cryptography may not be necessary in this context, and hardware real estate may be better utilized for other features.

One final question we must address is the flexibility of the hardware to user configuration. If a BLINK decoder is capable of various primitives like shifting, descrambling and watermarking, an encrypted footer already has some flexibility in commanding the device. There may be value in making this flexible enough that more fine-grained commands be passed through the display line, essentially rendering the BLINK decoder scriptable. We do not want to compromise the security of the system in the addition of features, however.

6. REFERENCES

[1] Altera Corporation. Avalon interface specifications. `http://www.altera.com/literature/manual/mnl_avalon_spec.pdf`, April 2009.

[2] R. Anderson. *Security Engineering: A Guide to Building Dependable Distributed Systems*. John Wiley and Sons, 2008.

[3] BITEC Ltd. Bitec hsmc dvi sdram loop-through reference design. `http://www.bitec.ltd.uk/hsmc_dvi_1024x768_c25.pdf`.

[4] S. Craver, Y. Chen, H. Chen, J. Yu, and I. Atakli. Blink: Securing information to the last connection. In *Proc. IEEE Conference on Consumer Communications and Networking*, January 2009.

[5] Digital Display Working Group. Digital visual interface. `http://www.ddwg.org/lib/dvi_10.pdf`.