Encryption-On-Demand:
Practical and Theoretical Considerations

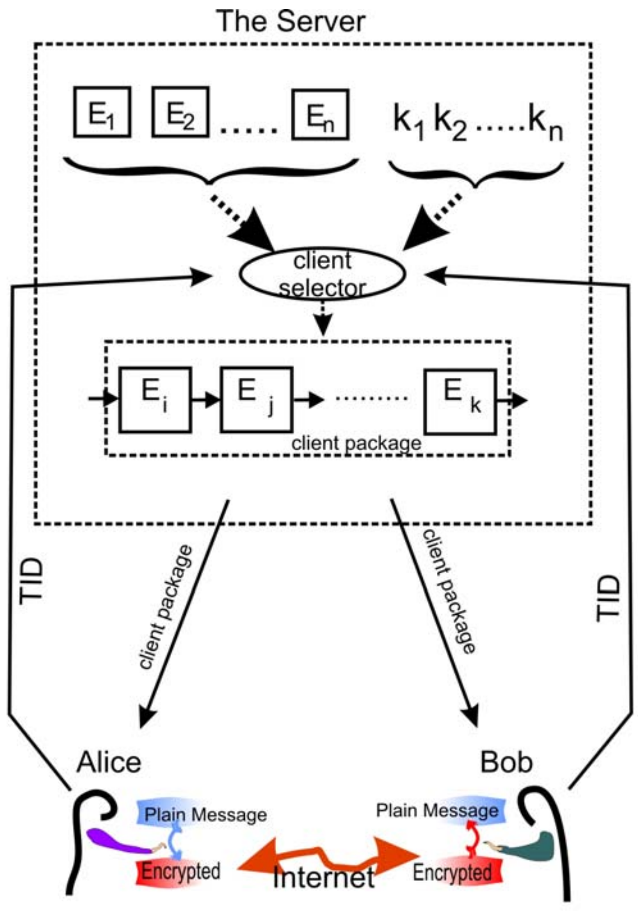
Gideon Samid
School of Engineering
Case Western Reserve University
Gideon.samid@case.edu

Alice and Bob may develop a spontaneous, yet infrequent need for online confidential exchange. They may be served by an 'encryption-on-demand' (EoD) service which will enable them to communicate securely with no prior preparations, and no after effects. We delineate a possible EoD service, and describe some of its theoretical and practical features. The proposed framework is a website which could tailor-make an encryption package to be downloaded by both Alice and Bob for their ad-hoc use. The downloaded package will include the cryptographic algorithm and a unique key, which may be of any size, since Alice and Bob will not have to enter, or regard the key per se, they would simply use the downloaded page to encrypt and decrypt their data. After their secure exchange both Alice and Bob may ignore, or discard the downloaded software, and restart the same procedure, with a different tailor-made package, exactly when needed. This framework allows for greater flexibility in managing the complexity aspects that ensures security. Alice and Bob will not have to know what encryption scheme they use. The server based tailoring program could pseudo-randomly pick AES, DES, RSA, ECC, select a short, or long key, and otherwise greatly increase the variability that would have to be negotiated by a cryptanalyst. Encryption-on-demand is offered on http://youdeny.com . Features are described.

1. INTRODUCTION: The online email community, which is fast spreading to include the majority of humanity, is by and large shunning encryption offerings, and rather communicating in plain language, reckoning that the vast majority of their communication is too lame, too ordinary, too uninteresting for anyone to track and infer upon. Whether this impression is true or not, it is prevailing, and only on rare occasions two or more online communication partners experience the need to use encryption and safeguard their exchange. Because of the low frequency of such requirement, most people don't prepare, don't acquire encryption capability, don't own a personal cryptographic key, and what is more, have little interest to burden their email program or document processor with a crypto-add-on. They are also quite reluctant to spend time familiarizing themselves with an obtuse protocol to follow, key management, security accounting and suchlike.

This reality may be well served by a service of encryption-on-demand, EoD.

1.1. ENCRYPTION-ON-DEMAND OUTLINED: The hub of the encryption-on-demand service is an accessible website fitted with an encryption tailoring capability (The encryption-on-demand server). Upon request, the website would tailor-make an encryption program, the 'client' program, deliver it to the requesting client, for his or her use. The client program will be uniquely identified by a tailoring id (TID). The same, or a different Internet surfer (client) would be able to forward the same TID to the server and receive the very same tailor-made copy. Hence, Alice and Bob may agree on a TID and then both would download the same copy to their personal computers. The TID agreement may be done a-priori, or ad-hoc using a different channel of communication. For example, Alice and Bob may exchange the TID over the phone, or instant messaging, or SMS. Since most of such privacy needs are to be experienced between close acquaintances, it should be quite easy for Alice and Bob to agree on a TID by referencing a piece of information they both know, but strangers don't. Such may be, birth city, date of birth, house address, etc. While the TID may look like a normal encryption key, it is really a different entity as outlined ahead.



The server 'cooks' a client package to download to Alice and Bob. It is a pseudo-random combination of all valid encryption systems, with choice keys, configured into encryption-on-demand package to allow Alice and Bob to communicate securely.

With Alice and bob in possession of identical client copies, they can now exchange any messages through email, instant messaging, SMS, etc. The protocol is quite simple. Alice types or pastes into the client package, the secret plain message she intends for Bob. She activates it, and the corresponding ciphertext is displayed. Alice might be oblivious to the logic and processing that worked on her plaintext. She needs not to furnish any key, or any identifier. Once the ciphertext is displayed, Alice copies it to the communication channel with Bob, say an email. Bob, upon receipt of the ciphertext, copies it to his client encryption software, activates it for decryption, and then faces the plaintext written by Alice. Like Alice, Bob has no clue, and no need to know what algorithm processed his ciphertext. He gets the plaintext clearly displayed and ready for consumption.

This protocol, will work in return, for Bob to write securely to Alice. The two can repeat such secure conversation for as long as they please, and when done, they may either keep their copy for future secure conversation or they may erase their copy to prevent anyone from stealing or confiscating their encryption/decryption program. When the need arises again for Alice and Bob to communicate with privacy then they may agree on another TID number, download its corresponding client package and repeat the same procedure.

1.1.1. THE TID vs. AN ENCRYPTION KEY: An encryption key is defined with respect to an unambiguous algorithm that would use the key to either transform the plaintext to its ciphertext, or vice versa. The TID, by contrast, serves just as a reference pointer to insure that Alice and Bob download the same piece of software. The server that uses this TID has great flexibility in determining what that downloaded client software will be comprised of. The server may be using every encryption scheme known to man, and string a software package that uses any number of these schemes in succession, say, through a pseudo random algorithm. Alice and Bob could not care less. They use the downloaded client by inputting either the plaintext, or the ciphertext, and outputting the opposite form. Alas, the cryptanalyst will face all that added variety which the server can use in setting up the tailored version. What is more, having concluded their ad-hoc communication, Alice and Bob can discard their client copy, and so even if the cryptanalyst invades their computer he would not be the wiser.

2. ENVIRONMENT AND PROCESS:

The environment of concern features Alice and Bob, two online surfers with occasional need for secure communication. It features a server site that offers encryption-on-demand services.

The server offers the following service: any Internet surfer may request a fully autonomous and standalone encryption package (the client software). The server defines a range of selection numbers: 1 to S, with S sufficiently large to make it infeasible for harry the hacker to guess the value of $1 \leq s \leq S$ picked by Alice, or Bob. We may prescribe for the server to produce S distinct client packages, so that no two package selection numbers $s_1 \neq s_2$ will encrypt the same plaintext p to the same ciphertext c, or vice versa.

The client package is complete, in the sense that it allows a user to enter a plaintext, p, and receive a corresponding ciphertext c, and vice versa. It contains the algorithms and the keys in a working single unit. This configuration allows the server to tailor the client package from a much larger set of possible clients than is the tailoring selection set of size S. The server set, $\mathbf{S}_0$, may remain a secret, and might change and vary as the server sees fit. This server flexibility would allow it to keep up with observed cryptanalysis capabilities. As the latter develop, so would the server's encryption.

2.1. THE SERVER: The server will be using any number of encryption systems: E_1, E_2, E_3, ...E_n and for each such system, the server would consider a full range of keys. The server might string these encryption schemes in any desired configuration, and in fact re-encrypt the output of one system with another, or perhaps another copy of the same encryption system, using the same or a different key. The full range of possibilities, and the exact algorithm for how the server uses the tailoring number to configure a particular client package may remain a secret held by the server without any need for it to be exposed to the using clients.

2.2. APPLICATION INTEGRATION: The encryption-on-demand protocol was described above as a standalone operation. Alice and Bob access their identical copy of the client software and generate either the ciphertext or the plaintext from the opposite form. However, the same protocol can be applied by integrating the client software to any document processing software used by Alice or Bob. Such would be mail programs, word processor, spread sheets, etc. Such integration would allow Alice and Bob to seamless communicate with security while each of them only sees the plaintext.

3. VISIBILITY AND CRYPTANALYSIS:

The encryption-on-demand protocol removes the burden and flexibility of choosing an encryption key from the actual users of the encryption capability. Alice and Bob will have a tailored encryption software with the encryption key fitted in. Since they can always restart the same operation with another client package, they might choose to erase and destroy the package they used before. Or they might lose it for that matter. Since Alice and bob don't pick keys, they also don't have to manage any keys.

The server entity may have complete visibility over the client package fitted with the key since the server 'cooked' that client package and sent it over to Alice and Bob. If then the server intercepts the communicated ciphertext, it can readily extract the plaintext. This analysis can be modified under two conditions:

- **1. anonymous download.**
- **2. multitude of clients.**

The first case is when Alice and Bob approach the server anonymously, say from a public computer. The server sends off the client, which Alice and Bob copy to their private storage device then use it on their own computer. The server would not know the identity of Alice and Bob. The second case refers to the situation where the encryption-on-demand protocol becomes popular and at given interval of time there are too numerous downloads to work with. In both cases, it would be a bit more difficult for the server to break the intercepted communication, but not insurmountably so.

The interesting aspect of this encryption-on-demand protocol is that the outsider cryptanalyst (not the server) would have a much more daunting task to break the ciphertext. The greater the variety of the server's options to configure clients packages, the more difficult is it for the cryptanalyst to crack the code. Normally it as assumed that the cryptanalyst is aware of the system used, the exact algorithm and anything else except the cryptographic key. This assumption definitely does not hold with this encryption-on-demand protocol. The cryptanalyst will have to consider the full range of configuration possibilities handled by the server. That means that the cryptanalyst will have to break in the server security because the server does not communicate its configuration variability -- it does not have to. So as long as the server team keeps the server facility secure, the cryptanalyst will find it nearly impossible to crack the ciphertext used by Alice and bob.

The fact that the server has cryptanalysis visibility while nobody else does, may suggest for the server to be run by an agency of authority, be it a corporate management, dispensing encryption clients to its people, or be it the government offering this service. In the latter case the government will insure that other governments, terrorists, and criminal organizations will be in the dark, while it enjoys operational visibility.

The described protocol, suffers, of course, from a weak link in the very communication of the client to Alice and Bob. Hackers can intercept the client, and use it to read the information traffic between Alice and Bob. This weak link can be handled either by sending the client via a different channel, or by employing standard asymmetric security protocols routinely used for commercial transactions.

4. APPLICATIONS:

The encryption-on-demand protocol could be used for self encrypting files, for communication between and among acquaintances, as well as among strangers. It could be used for authentication, for subliminal communication. The latter will be through using several client packages, and one that is being actually used signals something to the recipient.

Alice and Bob could amplify their security by deciding to use the encryption-on-demand protocol through two or more successive applications (where the ciphertext from one cycle becomes the plaintext for the next cycle). They can decide to combine two or more clients for one message, thereby further complicating the work of a cryptanalyst.

This encryption-on-demand concept is in line with the trend known as 'cloud computing' or web-based computing. Alice and Bob don't need to worry whether they have a robust enough encryption software. They trust the server to keep their client software up-to-date, and robust enough against any present danger of cryptanalysis. Otherwise Alice and Bob have to keep up-to-date, and upgrade their software on their own.

Encryption-on-demand is currently offered by YouDeny.com.

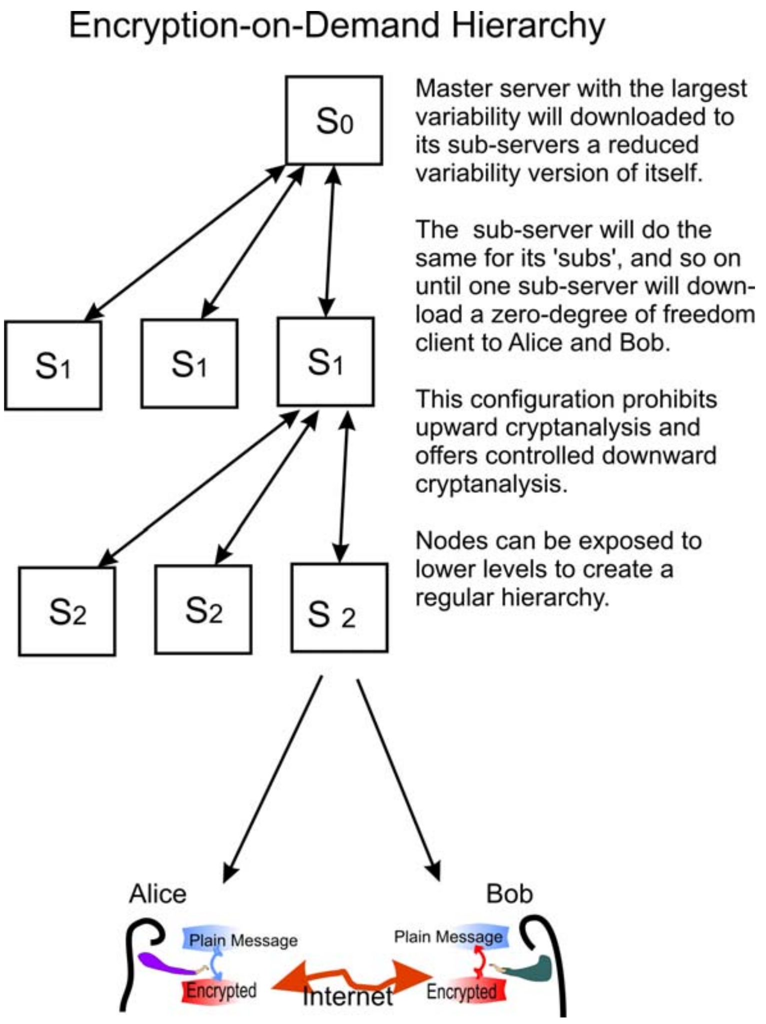
4.1. YouDeny.com: This active site implements the above described encryption-on-demand. Alice and Bob may choose a TID in the range of 1-999999999. The client software they receive in return is a full, autonomous, TID-tailored implementation of the Samid cipher, US Patent 6,823,068. The client package they receive appears as locally saved browser page with both the encryption and decryption algorithm implemented in JavaScript and completely transparent by using the "view source" option on the browser. The package features a message window and an encryption window. The user types or pastes the plain message onto the message window, clicks "Encrypt!" and the ciphertext appears in the encryption window. The user could then copy and paste the ciphertext to any email, or instant message stream, sending it to its destination. His or her partner, will copy the ciphertext from the incoming email or instant message, and paste it into the encryption window, click "Un-Encrypt!" and read the plaintext in the message window. This would conclude the secure communication. The two users may discard their YouDeny copy, and download another one for any subsequent secure communication.

The YouDeny server tailors for its users a customized unique instance of the Samid cipher. The Samid cipher has a variable size key, and a cryptanalyst must examine all key sizes in attempting to crack the cipher. The users, on their part could request YouDeny to send them a tailored copy that would link the sent ciphertext to an innocuous plaintext, in case they are under pressure to reveal the content of their encrypted message.

5. HIERARCHICAL EXTENSION:

In the basic setup described above Alice and Bob receive from the server a "fully cooked" client. We suppose now that Alice and Bob are given some latitude in deciding the exact makeup of their client package. That latitude is in the form of a selection key, k. In that case the client package downloaded from the server would have K degrees of freedom, and Alice and Bob when they pick a selection value $0 \leq k \leq K$ they eliminate this degree of freedom. We may regard any entity with some degrees of freedom as a subserver. In other words, the setup is modified by replacing Alice and Bob with 'sub-servers' who send the server a tailoring identifier, and receive in return a limited version of the server, with a smaller degree of freedom. Such a sub-server could work with Alice and Bob much the same way as the original server. But, in turn it could spawn a sub-sub-server, with a smaller variability of encryption configuration, and that sub-sib-server would either spawn its own 'sub' or work directly with Alice and Bob. We thus define a hierarchy.

In the basic setup the server can readily cryptanalyze the ciphertext exchanged between Alice and Bob because they have zero degree of freedom. However, two subservers may communicate securely by picking the same tailoring identifier. The server that downloaded to those subservers will be able to cryptanalyze the subserver communication by brute force trying all the possible copy selection options, that the server downloaded to the sub. This builds a visibility gradient whereby a parent node may control its ability to read its children's communication. And the higher up the node the less visibility thereto.


Encryption-on-Demand Hierarchy
S0
S1
S1
S1
S2
S2
S 2
Master server with the largest variability will downloaded to its sub-servers a reduced variability version of itself.
The sub-server will do the same for its 'subs', and so on until one sub-server will download a zero-degree of freedom client to Alice and Bob.
This configuration prohibits upward cryptanalysis and offers controlled downward cryptanalysis.
Nodes can be exposed to lower levels to create a regular hierarchy.
Alice
Bob
Plain Message
Encrypted
Internet
Plain Message
Encrypted

Reference:

Samid, G. 2001 "Re-Dividing Complexity Between Algorithms and Keys (Key Scripts)" The Second International Conference on Cryptology in India, Indian Institute of Technology, Madras, Chennai, India. December 2001. " Samid, G. 2002 " At-Will Intractability Up to Plaintext Equivocation Achieved via a Cryptographic Key Made As Small, or As Large As Desired - Without Computational Penalty " 2002 International Workshop on CRYPTOLOGY AND NETWORK SECURITY San Francisco, California, USA September 26 -- 28, 2002

Samid, G. 2001 "Anonymity Management: A Blue Print For Newfound Privacy" The Second International Workshop on Information Security Applications (WISA 2001), Seoul, Korea, September 13-14, 2001 (Best Paper Award).

Samid, G. 2005 "The Myth of Invincible Encryption" Digital Transactions May-June 2005