

The Paillier Cryptosystem

Andreas Steffen

Hochschule für Technik Rapperswil

andreas.steffen@hsr.ch

Agenda

- Some mathematical properties
- Encryption and decryption
- Additive homomorphic properties
- Zero knowledge proof for n-th powers
- Paillier e-voting simulator
- Non-interactive ZKP using the Fiat-Shamir heuristic
- Damgård-Jurik Cryptosystem (Generalized Paillier)
- Damgård-Jurik JavaScript e-voting client
- Threshold decryption schemes

The Paillier Cryptosystem I

Proposed by Pascal Paillier in 1999:

- Choose two large prime numbers p and q and form the modulus

$$n = pq$$

- Euler's totient function gives the number of elements in Z_n^*

$$\varphi(n) = (p-1)(q-1)$$

- The number of elements in $Z_{n^2}^*$ is

$$\varphi(n^2) = n\varphi(n)$$

- The **private key** λ is determined using Carmichael's function

$$\lambda(n) = \text{lcm}(p-1, q-1)$$

- Due to Carmichael's theorem, for every element $\omega \in Z_{n^2}^*$

$$\begin{cases} \omega^\lambda = 1 \pmod{n} \\ \omega^{n\lambda} = 1 \pmod{n^2} \end{cases}$$

The Paillier Cryptosystem II

- The **hard problem**: Deciding n -th composite residuosity!

$$z = y^n \pmod{n^2}$$
- The set of n -th residues is a multiplicative subgroup of $Z_{n^2}^*$ of order $\varphi(n)$
- Each n -th residue z has exactly n roots of degree n , among which exactly one is strictly smaller than n , namely

$$r = \sqrt[n]{z} \pmod{n}, \quad r \in Z_n^*$$
- The n -th roots of unity are the numbers of the form

$$(1+n)^m = 1 + mn \pmod{n^2}, \quad m \in Z_n$$
- Generate the multiplicative subgroup $Z_{n^2}^*$ as $Z_n \times Z_n^* \mapsto Z_{n^2}^*$

$$(m, r) \mapsto g^m \cdot r^n \pmod{n^2} = c$$

Paillier Encryption

m : plaintext message, r : random number for semantic security

Example: Multiplicative Subgroup $Z_{15^2}^*$

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	m
	1	16	31	46	61	76	91	106	121	136	151	166	181	196	211	g^m
1	1	16	31	46	61	76	91	106	121	136	151	166	181	196	211	
2	143	38	158	53	173	68	188	83	203	98	218	113	8	128	23	
4	199	34	94	154	214	49	109	169	4	64	124	184	19	79	139	
7	118	88	58	28	223	193	163	133	103	73	43	13	208	178	148	
8	107	137	167	197	2	32	62	92	122	152	182	212	17	47	77	
11	26	191	131	71	11	176	116	56	221	161	101	41	206	146	86	
13	82	187	67	172	52	157	37	142	22	127	7	112	217	97	202	
14	224	209	194	179	164	149	134	119	104	89	74	59	44	29	14	

r r^n

$$p = 3, q = 5, n = 15, n^2 = 225$$

$$\varphi(n) = 8, \lambda(n) = \text{lcm}(2, 4) = 4$$

Generator in most general form:

$$g = (1 + \alpha \cdot n)\beta^n, \quad \alpha, \beta \in Z_n^*$$

Paillier Decryption

$$m = \frac{L(c^\lambda \bmod n^2)}{L(g^\lambda \bmod n^2)} \bmod n \quad \text{with} \quad L(x) = \frac{x-1}{n}$$

- Apply the private key λ and use Carmichael's theorem

$$c^\lambda = (g^m \cdot r^n)^\lambda = g^{m\lambda} \cdot r^{n\lambda} = g^{m\lambda}$$

- Make use of the relationship $(1+n)^x = 1 + xn \bmod n^2$

$$g^{m\lambda} = ((1+n)^\alpha \beta^n)^{m\lambda} = (1+n)^{\alpha\lambda m} \beta^{n\lambda m} = (1 + \alpha\lambda mn) \bmod n^2$$

- Apply the $L(x)$ function

$$\frac{L(1 + \alpha\lambda mn)}{L(1 + \alpha\lambda n)} = \frac{\alpha\lambda m}{\alpha\lambda} \bmod n = m$$

Additive Homomorphic Properties

$$D(E(m_1) \cdot E(m_2) \bmod n^2) = m_1 + m_2 \bmod n$$

- Verification

$$E(m_1) \cdot E(m_2) = g^{m_1} r_1^n \cdot g^{m_2} r_2^n = g^{m_1+m_2} \cdot r_1^n r_2^n \bmod n^2$$

$$D(E(m)^k \bmod n^2) = k \cdot m \bmod n$$

- Use in e-voting systems with homomorphic tallying:

The **additive** homomorphic property directly returning the tally is the biggest advantage of the **Paillier Cryptosystem** over the **El Gamal Cryptosystem** which has an intrinsically **multiplicative** homomorphic property requiring the computation of a discrete logarithm over a bounded range to extract the tally.

Validity Proof of Ballot (Case: $k = i$)

- K valid voting messages (e.g. vote for one out of K candidates)

$$m_1, m_2, \dots, m_k, \dots, m_K$$

- Zero knowledge proof : Prove that u_k is an n -th power

$$u_k = \frac{c}{g^{m_k}} = \frac{g^{m_i} \cdot r^n}{g^{m_k}} \bmod n^2 = r^n \quad \text{only if } m_i = m_k$$

- Commitment: Prover chooses a random number ω

$$a_i = \omega^n \bmod n^2, \quad \omega \in \mathbb{Z}_n^*$$

- Challenge: Verifier chooses a random bit string e_i of length b

$$e_i < 2^b, \quad 2^b < \min(p, q)$$

- Response: Prover computes z_i

$$z_i = \omega \cdot r^{e_i} \bmod n$$

- Verification: $z_i^n = a_i \cdot u_i^{e_i} \bmod n^2$

$$z_i^n = (\omega \cdot r^{e_i})^n = \omega^n \cdot r^{n \cdot e_i} \bmod n^2$$

Validity Proof of Ballot (Cases: $k \neq i$)

- Preparation: Prover chooses z_k and bit string e_k randomly

$$z_k \in \mathbb{Z}_n^*, \quad e_k < 2^b, \quad 2^b < \min(p, q)$$

- Commitment: Prover computes a_k so that it passes verification

$$a_k = \frac{z_k^n}{u_k^{e_k}} \bmod n^2$$

- Challenge: Verifier chooses a random bit string e of length b

$$e < 2^b, \quad 2^b < \min(p, q)$$

- Response: Prover sends prepared z_k and e_k

- Verification: $z_k^n = a_k \cdot u_k^{e_k} \bmod n^2$

$$\sum_{k=1}^K e_k = e \bmod 2^b$$

Prover can preselect all e_k for $k \neq i$ but is bound by e for the choice of e_i .

Paillier E-Voting Simulator

Paillier Cryptosystem - Mozilla Firefox

File Edit View History Bookmarks Tools Help

http://security.hsr.ch/msevot/paillier?p=3&q=11&v1=1&m1=1&v

Paillier Cryptosystem

Paillier Cryptosystem

RSA Factors: p = 3 q = 11 Start

Voter	Candidates	Cheat
V1	<input type="radio"/> None <input checked="" type="radio"/> C1 <input type="radio"/> C2 <input type="radio"/> C3	1
V2	<input type="radio"/> None <input type="radio"/> C1 <input checked="" type="radio"/> C2 <input type="radio"/> C3	1
V3	<input type="radio"/> None <input type="radio"/> C1 <input checked="" type="radio"/> C2 <input type="radio"/> C3	1
V4	<input type="radio"/> None <input type="radio"/> C1 <input type="radio"/> C2 <input checked="" type="radio"/> C3	2
V5	<input type="radio"/> None <input type="radio"/> C1 <input checked="" type="radio"/> C2 <input type="radio"/> C3	1
V6	<input type="radio"/> None <input type="radio"/> C1 <input type="radio"/> C2 <input checked="" type="radio"/> C3	1
V7	<input type="radio"/> None <input checked="" type="radio"/> C1 <input type="radio"/> C2 <input type="radio"/> C3	2
V8	<input checked="" type="radio"/> None <input type="radio"/> C1 <input type="radio"/> C2 <input type="radio"/> C3	1
Tally	1 1 3 1	2

Hide details of:

Paillier Encryption / Decryption

Zero-Knowledge Proof

Generator g:

α fixed to 1

β fixed to 1

Message Base

Random Seed

$p = 3, q = 11, n = pq = 33, n_2 = 1089, \phi(n) = (p-1)(q-1) = 20, \lambda(n) = \text{lcm}(p-1, q-1) = 10$

Valid Voting Messages: $m = m_i, \text{None: } m_0 = 0, C1: m_1 = 1, C2: m_2 = b = 4, C3: m_3 = b^2 = 16$

Done

E-Voting Simulator – Tallying with ZKPs

Paillier Cryptosystem - Mozilla Firefox

File Edit View History Bookmarks Tools Help

<http://security.hsr.ch/msevotepaillier?p=3&q=11&v1=1&m1=1&v2=2&m2=18>

Paillier Cryptosystem

Voting and Tallying: $t = \Pi(c[m, r]) \text{ mod } n^2$

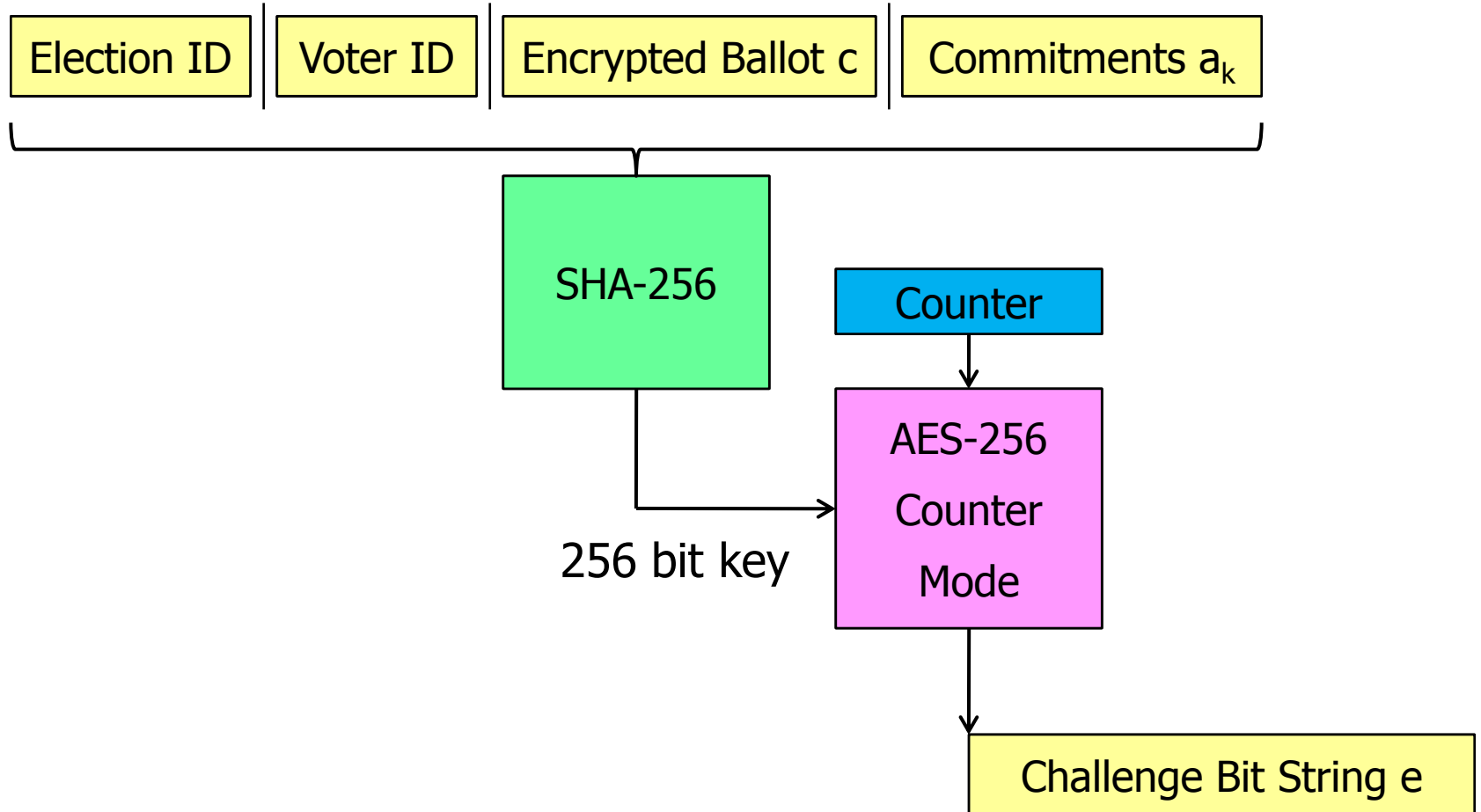
Voter	m	r	c	u0	u1	u2	u3	a0	a1	a2	a3	es	e0	e1	e2	e3	w	z0	z1	z2	z3	z0^n	z1^n	z2^n	z3^n	t	tm	C1	C2	C3
V1	1	26	911	911	251	449	152	346	602	856	215	0	1	1	0	0	2	23	19	4	8	485	820	856	215	911	1	1	0	0
V2	4	5	488	488	719	323	917	400	379	971	766	0	1	1	0	0	20	14	26	20	28	269	251	971	766	256	5	1	1	0
V3	4	20	641	641	179	971	872	602	856	928	881	0	0	0	1	1	16	2	4	23	31	602	856	485	487	746	9	1	2	0
V4	32	26	680	680	20	218	1010	745	485	526	485	1	1	0	1	1	23	8	23	5	4	215	485	323	856	746	9	1	2	0
V5	4	28	601	601	370	766	172	31	98	485	971	0	1	1	0	0	23	13	5	23	20	118	323	485	971	767	13	1	3	0
V6	16	31	619	619	883	586	487	896	568	701	604	1	1	1	1	0	10	5	10	29	10	323	604	233	604	1058	29	1	3	1
V7	2	8	248	248	776	182	1073	928	766	490	838	0	0	1	1	0	28	16	26	20	7	928	251	971	838	1058	29	1	3	1
V8	0	31	487	487	751	454	355	251	1088	133	862	0	0	0	1	1	26	26	32	31	1	251	1088	487	1	149	29	1	3	1

The source code of the simulator is available under a GPLv2 license.

© 2008-2009 by Andreas Steffen, HSR Hochschule für Technik Rapperswil

Done

Non-Interactive ZKP (Fiat-Shamir Heuristic)



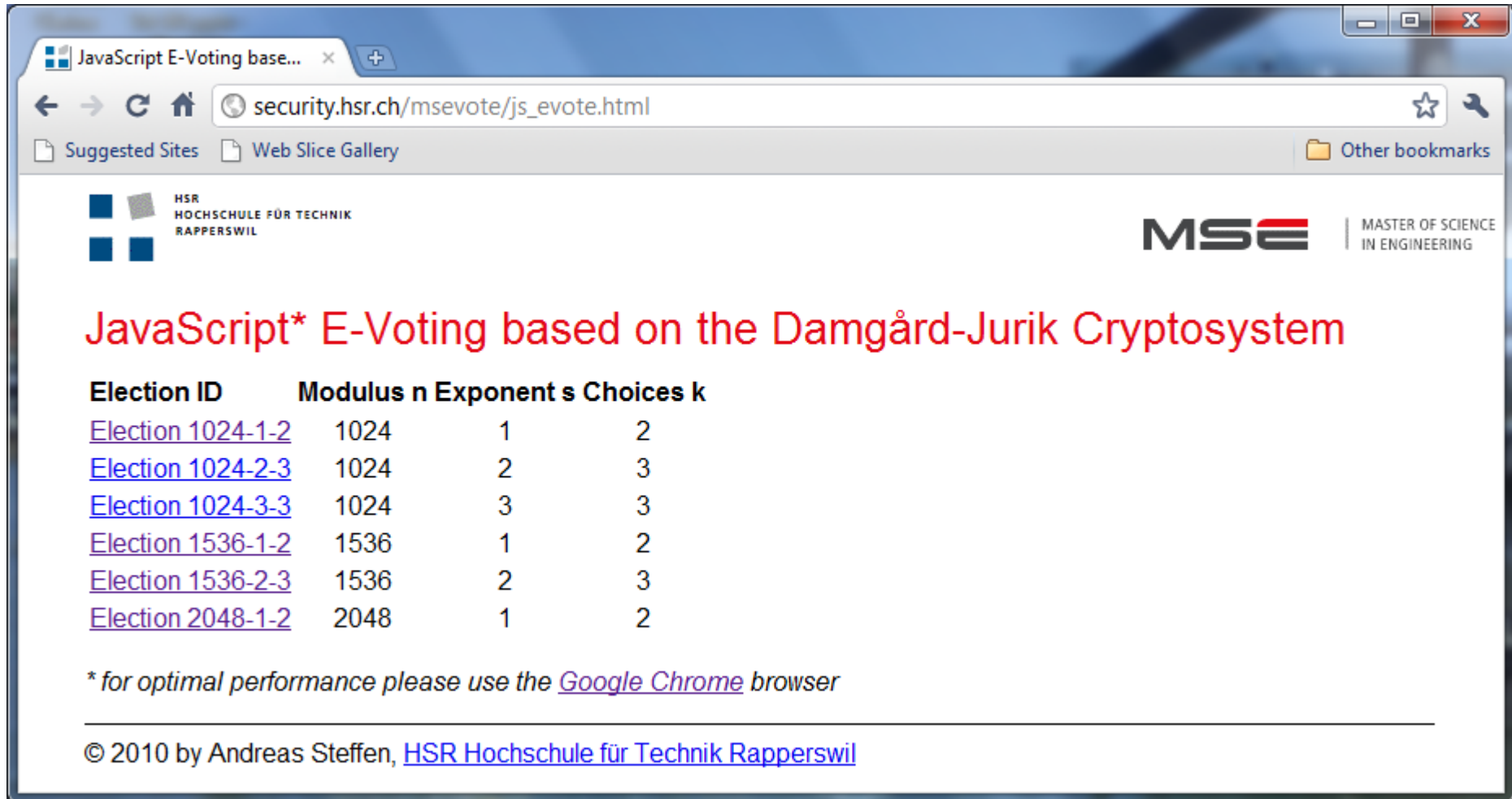
The Damgård-Jurik Cryptosystem

- Additional parameter s (Paillier: $s = 1$)

$$(m, r) \mapsto g^m \cdot r^{n^s} \bmod n^{s+1} = c$$

m : plaintext message, r : random number for semantic security

- Generate the multiplicative subgroup $Z_{n^{s+1}}^*$ as $Z_{n^s} \times Z_n^* \mapsto Z_{n^{s+1}}^*$
- Generator usually chosen as $g = (1+n)$
- Size of modulus n : b bits (e.g. 1536 bits)
- Size of message m : $s \cdot b - 1$ bits ($s=1$: 1535 bits, $s=2$: 3071 bits)
- Size of ciphertext c : $(s+1) \cdot b$ ($s=1$: 3072 bits, $s=2$: 4608 bits)
- Efficiency: $\mu = s/(s+1)$ ($s=1$: 50%, $s=2$: 67%, $s=3$: 75%)



The screenshot shows a web browser window with the address bar containing `security.hsr.ch/msevot/js_evote.html`. The page header includes the HSR Hochschule für Technik Rapperswil logo on the left and the MSE Master of Science in Engineering logo on the right. The main content area features a red heading: "JavaScript* E-Voting based on the Damgård-Jurik Cryptosystem". Below this heading is a table with four columns: "Election ID", "Modulus n", "Exponent s", and "Choices k". The table lists six election entries with their respective parameters. A note below the table states: "* for optimal performance please use the [Google Chrome](#) browser". At the bottom of the page, there is a copyright notice: "© 2010 by Andreas Steffen, [HSR Hochschule für Technik Rapperswil](#)".

Election ID	Modulus n	Exponent s	Choices k
Election 1024-1-2	1024	1	2
Election 1024-2-3	1024	2	3
Election 1024-3-3	1024	3	3
Election 1536-1-2	1536	1	2
Election 1536-2-3	1536	2	3
Election 2048-1-2	2048	1	2

** for optimal performance please use the [Google Chrome](#) browser*

© 2010 by Andreas Steffen, [HSR Hochschule für Technik Rapperswil](#)

Damgård-Jurik JavaScript E-Voting Client

JavaScript E-Voting base... x E-Voting Webclient - Pro... x

security.hsr.ch/msevot/js_evote-1536-2-3.html

Suggested Sites Web Slice Gallery Other bookmarks

HSR HOCHSCHULE FÜR TECHNIK RAPPERSWIL

E-Voting Web-Client in JavaScript
S. Samkang & P. Dorigo

e voting

VOTER ID

Voter-ID:

ELECTION 1536-2-3

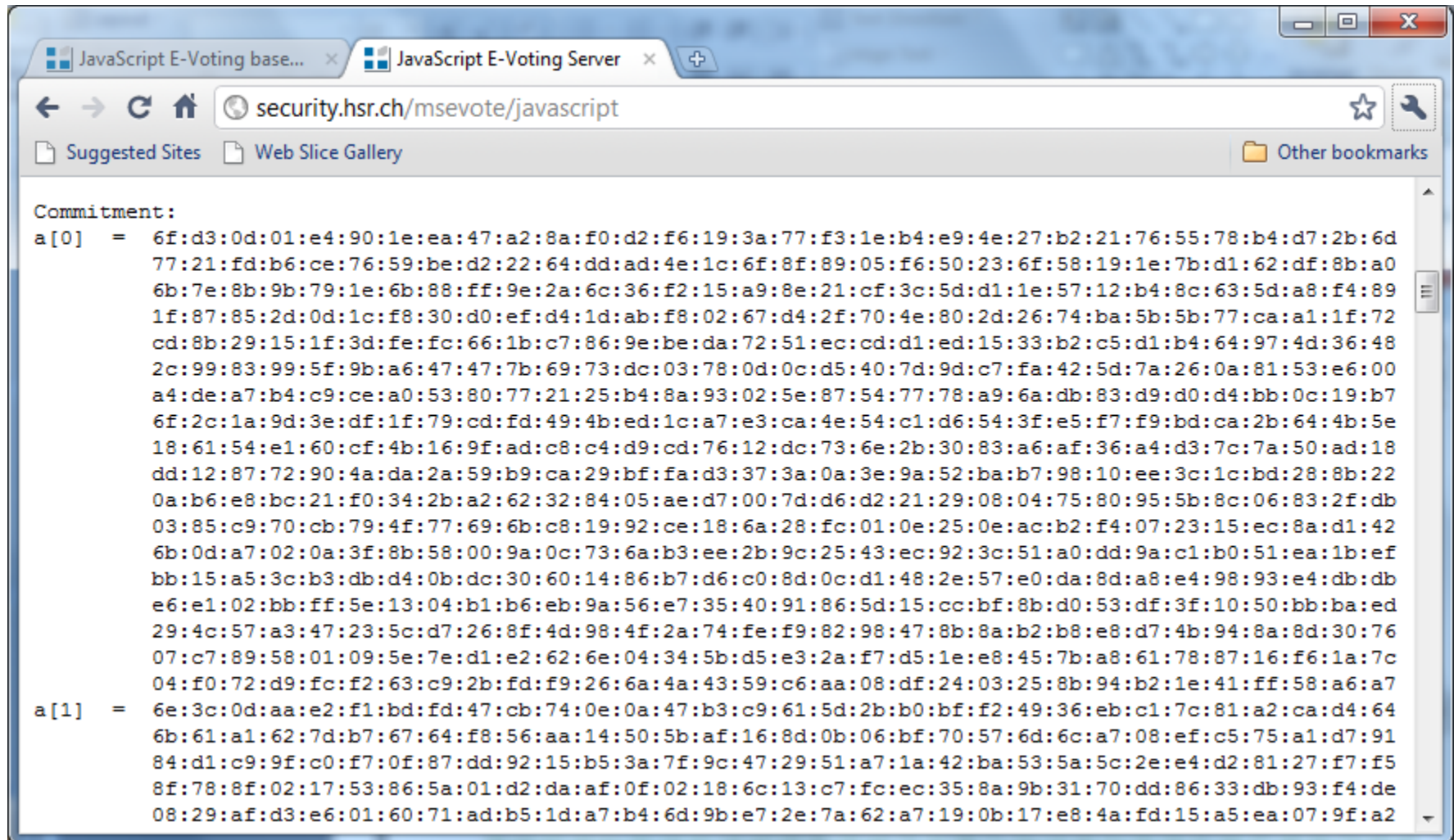
Not initialized

Please vote for your favorite candidate:

Alice Bob Carol None Cheat

Encrypt

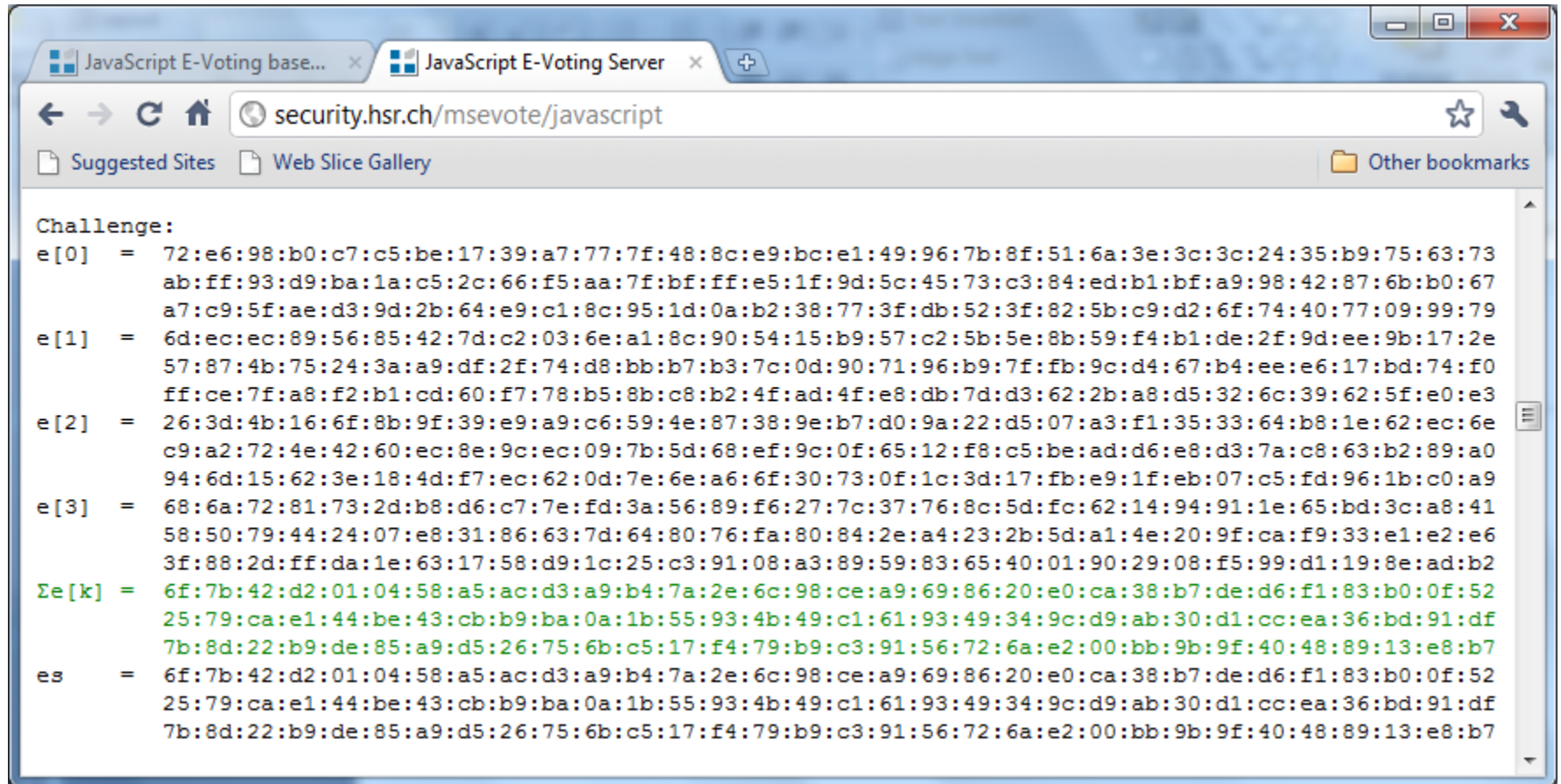
Commitment



The screenshot shows a web browser window with two tabs: "JavaScript E-Voting base..." and "JavaScript E-Voting Server". The address bar displays "security.hsr.ch/msevote/javascript". Below the address bar, there are "Suggested Sites" and "Web Slice Gallery" on the left, and "Other bookmarks" on the right. The main content area displays the following text:

```
Commitment:  
a[0] = 6f:d3:0d:01:e4:90:1e:ea:47:a2:8a:f0:d2:f6:19:3a:77:f3:1e:b4:e9:4e:27:b2:21:76:55:78:b4:d7:2b:6d  
77:21:fd:b6:ce:76:59:be:d2:22:64:dd:ad:4e:1c:6f:8f:89:05:f6:50:23:6f:58:19:1e:7b:d1:62:df:8b:a0  
6b:7e:8b:9b:79:1e:6b:88:ff:9e:2a:6c:36:f2:15:a9:8e:21:cf:3c:5d:d1:1e:57:12:b4:8c:63:5d:a8:f4:89  
1f:87:85:2d:0d:1c:f8:30:d0:ef:d4:1d:ab:f8:02:67:d4:2f:70:4e:80:2d:26:74:ba:5b:5b:77:ca:a1:1f:72  
cd:8b:29:15:1f:3d:fe:fc:66:1b:c7:86:9e:be:da:72:51:ec:cd:d1:ed:15:33:b2:c5:d1:b4:64:97:4d:36:48  
2c:99:83:99:5f:9b:a6:47:47:7b:69:73:dc:03:78:0d:0c:d5:40:7d:9d:c7:fa:42:5d:7a:26:0a:81:53:e6:00  
a4:de:a7:b4:c9:ce:a0:53:80:77:21:25:b4:8a:93:02:5e:87:54:77:78:a9:6a:db:83:d9:d0:d4:bb:0c:19:b7  
6f:2c:1a:9d:3e:df:1f:79:cd:fd:49:4b:ed:1c:a7:e3:ca:4e:54:c1:d6:54:3f:e5:f7:f9:bd:ca:2b:64:4b:5e  
18:61:54:e1:60:cf:4b:16:9f:ad:c8:c4:d9:cd:76:12:dc:73:6e:2b:30:83:a6:af:36:a4:d3:7c:7a:50:ad:18  
dd:12:87:72:90:4a:da:2a:59:b9:ca:29:bf:fa:d3:37:3a:0a:3e:9a:52:ba:b7:98:10:ee:3c:1c:bd:28:8b:22  
0a:b6:e8:bc:21:f0:34:2b:a2:62:32:84:05:ae:d7:00:7d:d6:d2:21:29:08:04:75:80:95:5b:8c:06:83:2f:db  
03:85:c9:70:cb:79:4f:77:69:6b:c8:19:92:ce:18:6a:28:fc:01:0e:25:0e:ac:b2:f4:07:23:15:ec:8a:d1:42  
6b:0d:a7:02:0a:3f:8b:58:00:9a:0c:73:6a:b3:ee:2b:9c:25:43:ec:92:3c:51:a0:dd:9a:c1:b0:51:ea:1b:ef  
bb:15:a5:3c:b3:db:d4:0b:dc:30:60:14:86:b7:d6:c0:8d:0c:d1:48:2e:57:e0:da:8d:a8:e4:98:93:e4:db:db  
e6:e1:02:bb:ff:5e:13:04:b1:b6:eb:9a:56:e7:35:40:91:86:5d:15:cc:bf:8b:d0:53:df:3f:10:50:bb:ba:ed  
29:4c:57:a3:47:23:5c:d7:26:8f:4d:98:4f:2a:74:fe:f9:82:98:47:8b:8a:b2:b8:e8:d7:4b:94:8a:8d:30:76  
07:c7:89:58:01:09:5e:7e:d1:e2:62:6e:04:34:5b:d5:e3:2a:f7:d5:1e:e8:45:7b:a8:61:78:87:16:f6:1a:7c  
04:f0:72:d9:fc:f2:63:c9:2b:fd:f9:26:6a:4a:43:59:c6:aa:08:df:24:03:25:8b:94:b2:1e:41:ff:58:a6:a7  
a[1] = 6e:3c:0d:aa:e2:f1:bd:fd:47:cb:74:0e:0a:47:b3:c9:61:5d:2b:b0:bf:f2:49:36:eb:c1:7c:81:a2:ca:d4:64  
6b:61:a1:62:7d:b7:67:64:f8:56:aa:14:50:5b:af:16:8d:0b:06:bf:70:57:6d:6c:a7:08:ef:c5:75:a1:d7:91  
84:d1:c9:9f:c0:f7:0f:87:dd:92:15:b5:3a:7f:9c:47:29:51:a7:1a:42:ba:53:5a:5c:2e:e4:d2:81:27:f7:f5  
8f:78:8f:02:17:53:86:5a:01:d2:da:af:0f:02:18:6c:13:c7:fc:ec:35:8a:9b:31:70:dd:86:33:db:93:f4:de  
08:29:af:d3:e6:01:60:71:ad:b5:1d:a7:b4:6d:9b:e7:2e:7a:62:a7:19:0b:17:e8:4a:fd:15:a5:ea:07:9f:a2
```


Challenge Verification



```
Challenge:  
e[0] = 72:e6:98:b0:c7:c5:be:17:39:a7:77:7f:48:8c:e9:bc:e1:49:96:7b:8f:51:6a:3e:3c:3c:24:35:b9:75:63:73  
ab:ff:93:d9:ba:1a:c5:2c:66:f5:aa:7f:bf:ff:e5:1f:9d:5c:45:73:c3:84:ed:b1:bf:a9:98:42:87:6b:b0:67  
a7:c9:5f:ae:d3:9d:2b:64:e9:c1:8c:95:1d:0a:b2:38:77:3f:db:52:3f:82:5b:c9:d2:6f:74:40:77:09:99:79  
e[1] = 6d:ec:ec:89:56:85:42:7d:c2:03:6e:a1:8c:90:54:15:b9:57:c2:5b:5e:8b:59:f4:b1:de:2f:9d:ee:9b:17:2e  
57:87:4b:75:24:3a:a9:df:2f:74:d8:bb:b7:b3:7c:0d:90:71:96:b9:7f:fb:9c:d4:67:b4:ee:e6:17:bd:74:f0  
ff:ce:7f:a8:f2:b1:cd:60:f7:78:b5:8b:c8:b2:4f:ad:4f:e8:db:7d:d3:62:2b:a8:d5:32:6c:39:62:5f:e0:e3  
e[2] = 26:3d:4b:16:6f:8b:9f:39:e9:a9:c6:59:4e:87:38:9e:b7:d0:9a:22:d5:07:a3:f1:35:33:64:b8:1e:62:ec:6e  
c9:a2:72:4e:42:60:ec:8e:9c:ec:09:7b:5d:68:ef:9c:0f:65:12:f8:c5:be:ad:d6:e8:d3:7a:c8:63:b2:89:a0  
94:6d:15:62:3e:18:4d:f7:ec:62:0d:7e:6e:a6:6f:30:73:0f:1c:3d:17:fb:e9:1f:eb:07:c5:fd:96:1b:c0:a9  
e[3] = 68:6a:72:81:73:2d:b8:d6:c7:7e:fd:3a:56:89:f6:27:7c:37:76:8c:5d:fc:62:14:94:91:1e:65:bd:3c:a8:41  
58:50:79:44:24:07:e8:31:86:63:7d:64:80:76:fa:80:84:2e:a4:23:2b:5d:a1:4e:20:9f:ca:f9:33:e1:e2:e6  
3f:88:2d:ff:da:1e:63:17:58:d9:1c:25:c3:91:08:a3:89:59:83:65:40:01:90:29:08:f5:99:d1:19:8e:ad:b2  
Σe[k] = 6f:7b:42:d2:01:04:58:a5:ac:d3:a9:b4:7a:2e:6c:98:ce:a9:69:86:20:e0:ca:38:b7:de:d6:f1:83:b0:0f:52  
25:79:ca:e1:44:be:43:cb:b9:ba:0a:1b:55:93:4b:49:c1:61:93:49:34:9c:d9:ab:30:d1:cc:ea:36:bd:91:df  
7b:8d:22:b9:de:85:a9:d5:26:75:6b:c5:17:f4:79:b9:c3:91:56:72:6a:e2:00:bb:9b:9f:40:48:89:13:e8:b7  
es = 6f:7b:42:d2:01:04:58:a5:ac:d3:a9:b4:7a:2e:6c:98:ce:a9:69:86:20:e0:ca:38:b7:de:d6:f1:83:b0:0f:52  
25:79:ca:e1:44:be:43:cb:b9:ba:0a:1b:55:93:4b:49:c1:61:93:49:34:9c:d9:ab:30:d1:cc:ea:36:bd:91:df  
7b:8d:22:b9:de:85:a9:d5:26:75:6b:c5:17:f4:79:b9:c3:91:56:72:6a:e2:00:bb:9b:9f:40:48:89:13:e8:b7
```

Response Verification

The screenshot shows a web browser window with two tabs: "JavaScript E-Voting base..." and "JavaScript E-Voting Server". The address bar displays "security.hsr.ch/msevote/javascript". Below the address bar, there are "Suggested Sites" and "Web Slice Gallery" sections. The main content area shows a JavaScript console with the following output:

```
m[1] = 00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:01:00:00:00:00:00:00:00
z[1] = d4:e5:bf:f3:dc:55:4f:27:d7:cf:cf:a9:1d:77:94:bd:3c:99:4f:23:72:1c:e2:52:c1:c7:e7:b8:10:2a:36:ad
90:74:0c:83:e9:c6:36:4e:2d:f4:72:1a:5b:7a:cf:e4:e3:57:4a:6b:3d:b0:b8:6c:1d:91:2b:10:7c:2b:2d:73
59:bc:ce:1f:42:40:a9:fb:29:e7:a1:0c:5f:f4:6b:56:5a:60:be:e5:60:df:47:e7:60:0a:3c:dc:4c:b9:c7:ef
c2:37:d0:6f:64:b5:58:c5:94:b5:30:4d:41:74:04:ac:fb:9e:68:9b:55:be:be:0f:b4:91:8e:bd:e8:eb:0a:ae
e4:85:7b:1c:be:da:22:70:71:34:d7:80:a5:b0:63:15:eb:0d:da:e5:a5:a7:a6:ce:4a:c6:b1:be:d1:92:18:4d
96:56:75:5b:3f:33:00:9c:8b:3d:8b:dc:67:96:b7:f4:fc:2b:3e:b1:93:53:af:29:a5:bc:ef:22:e1:f5:bf:15
lh[1] = 2c:e8:ba:0c:f7:07:70:b7:05:eb:76:fe:7e:83:77:ab:e2:82:78:0f:b9:1f:97:d8:10:34:9d:cf:d0:03:96:14
ff:6c:ff:b5:52:0f:b2:10:77:fc:d9:cd:dc:49:5a:7a:d8:72:54:9f:99:78:0a:50:85:af:ab:2a:d9:8e:f1:ce
78:d8:6a:01:dc:fd:7e:d8:a6:2e:e2:d8:78:67:0f:80:22:e0:5c:e9:d8:41:75:7e:d2:01:1d:d1:fc:28:7d:12
b1:78:eb:e5:20:53:3b:1b:dd:4e:5e:e1:e5:93:70:bd:d4:3c:a3:29:3a:db:9c:06:66:6d:6e:96:06:40:2d:2c
6b:bf:24:f7:96:a1:81:e0:43:c8:9b:bb:a6:8a:44:38:dd:73:62:a8:ee:e2:d4:73:a4:81:f6:7d:ed:86:c6:9b
6c:ea:00:8c:8d:bd:06:b7:26:a4:1d:bb:60:87:c4:7f:a7:34:5c:9c:bf:2e:65:7b:18:2a:58:a7:bb:6b:e4:92
99:e2:f3:d8:d1:f4:4c:74:76:23:a2:73:04:08:49:7e:12:eb:15:1c:88:1e:0f:0b:8b:36:bb:96:3f:81:d0:bc
a6:e4:0c:f7:cb:08:28:ec:ca:3a:f7:91:f4:19:03:c7:c2:ea:59:9b:ca:6c:31:8c:f7:31:87:d3:0f:c1:79:dc
cf:d2:5e:66:7a:03:73:37:eb:c7:99:45:6f:5e:95:73:93:81:cc:12:b2:d4:b5:45:f3:60:c0:c0:10:86:06:86
84:f6:a2:2a:ec:36:4e:9c:67:76:03:e4:3e:ab:cb:ae:e9:63:42:05:1e:89:d1:88:e6:2a:c8:0a:5e:da:90:8f
b9:55:16:cd:40:cf:f2:e0:ab:2c:87:b7:2e:ca:a6:45:66:d0:6f:b0:94:7a:32:88:f0:73:77:9a:f5:43:a6:04
c9:c5:42:fc:7b:ef:ed:fe:32:69:a6:a2:53:fb:c6:61:c9:1e:27:b9:0f:c2:a2:ee:6d:1d:28:9d:f1:c5:cd:5e
4b:ad:d9:a6:e7:7b:14:fa:09:3d:70:17:a1:b9:14:10:41:4d:ac:c9:68:9e:a5:6a:e5:7d:2f:1b:9d:2c:21:3a
d9:12:ba:07:e3:1c:55:f9:db:12:7b:01:b8:c0:3d:ea:7a:80:a4:5f:d0:f7:b9:44:ca:21:66:7d:75:a2:d2:dc
cc:16:32:44:b8:28:4b:4a:26:ff:d8:d8:26:16:1a:be:11:c7:be:7d:ba:28:34:58:5c:ea:8b:7d:aa:b1:31:71
09:c5:b0:d3:bc:85:56:80:c6:ef:4f:d5:3f:ef:75:32:95:71:5a:16:d4:2c:35:06:41:96:f7:f6:6e:74:45:d7
01:f0:d4:52:66:4d:10:1b:f4:ff:b0:59:75:2c:cb:5e:6b:75:f5:67:4b:c0:ea:89:ac:b9:0e:a4:08:a4:4b:62
e5:af:6d:ba:49:77:e7:f4:d4:4c:97:70:b8:1e:69:51:d2:22:db:12:18:b6:f0:7f:64:2b:b9:a3:84:32:f6:5c
rh[1] = 2c:e8:ba:0c:f7:07:70:b7:05:eb:76:fe:7e:83:77:ab:e2:82:78:0f:b9:1f:97:d8:10:34:9d:cf:d0:03:96:14
ff:6c:ff:b5:52:0f:b2:10:77:fc:d9:cd:dc:49:5a:7a:d8:72:54:9f:99:78:0a:50:85:af:ab:2a:d9:8e:f1:ce
```

Threshold Scheme with a Trusted Dealer

Damgård-Jurik Cryptosystem - Mozilla Firefox

File Edit View History Bookmarks Tools Help

http://security.hsr.ch/msevote/damgardjurik?p=7&q=11&v1=1&m1=1t

Damgård-Jurik Cryptosystem

Secret Sharing: $N = 5, T = 3, \Delta = N! = 120, 1/(4\alpha\Delta^2) \bmod n^s = 4288$

Secret Sharing Polynomial: $d(x) = a_0 + a_1*x + a_2*x^2 \bmod n'*n^s = 65220 + 5438*x + 22261*x^2 \bmod 88935$

Partial Decryption: $c_i = t^{[2\Delta*d(i)]} \bmod n^{(s+1)}$

i	d(i)	c _i
A1	3984	146532
A2	76205	101641
A3	15078	148226
A4	87408	221068
A5	26390	450605

Lagrange Interpolation: $c' = \Pi(c_i^{[2*\lambda_i]}) / (4\alpha\Delta^2) \bmod n^{(s+1)}$

Iterative Mapping: $tm = I(c') * \mu \bmod n^s$

Authorities	λ_1	λ_2	λ_3	λ_4	λ_5	c'	tm
A1 & A2 & A3	360	-360	120			67761	55
A3 & A4 & A5			1200	-1800	720	67761	55
A1 & A2 & A4 & A5	400	-400		200	-80	67761	55
A1 & A2 & A3 & A4 & A5	600	-1200	1200	-600	120	67761	55

Done

- Practical threshold RSA signatures without a trusted dealer
Ivan Damgard, Maciej Koprowski, 2001
- The distributed generation of an RSA private key required by a **Threshold Paillier Cryptosystem** is much **more complex** than the **simple independent** partial private key generation possible with the **El Gamal Cryptosystem**.